REMARKS

Claims 1-44 are pending in this application.

Specification

The paragraph on page 5, lines 8-14, has been replaced to correct a typographical error. Applicant respectfully submits that no new matter has been added and requests that the Patent Office withdraw the objection to the specification.

Claim Rejections - 35 USC § 103

The Patent Office rejected claims 1-5, 7-11, 13-16, 18, and 20-44 under 35 USC 103(a) as being unpatentable over Blandy, U.S. Patent No. 6,295,642, in view of Coutts et al., U.S. Patent No. 6,311,165.

The present invention provides an execution environment for optimizing the efficiency of the distributed object system. In accordance with the present invention, a compiler with ability to interpret, a just in time compiler, and a pre-compiler may be implemented together. A dynamic base object is provided in a tagged file format that allows a compiler to identify critical sections of the object code for immediate one-time compilations while delaying compilation of non-tagged, non-critical code until required by the system.

When applying 35 U.S.C. 103, the following tenets of patent law must be adhered to: (A) the claimed invention must be considered as a whole; (B) the references must be considered as a whole and must suggest the desirability and thus the obviousness of making the combination; (C) the references must be viewed without the benefit of impermissible hindsight vision afforded by the claimed invention; and (D) reasonable expectation of success is the standard with which obviousness is determined. See MPEP § 2141 and Hodosh v. Block Drug Co., Inc., 786 F.2d 1136, 1143 n.5, 220 USPQ 182, 187 n.5 (Fed. Cir. 1986).

The Patent Office has based the rejection of all claims on a combination of Blandy and Coutts. In particular, the Patent Office has asserted that Blandy is deficient in not teaching "the independent byte-code includes at least one dynamic base object, the at least one dynamic base object comprising an interface dynamic base object and an implementation dynamic base object that communicate with each other over a message bus." However, Claims 13-17 and 35 do not recite this limitation. That is, were Coutts to provide a valid teaching of this limitation, Coutts would not be needed because Claims 13-17 and 35 lack the limitation Coutts has been alleged to teach. Claims 13-17 and 35 recite "a loader ... being capable of ... pre-compiling," "an identifier coupled to the loader, the identifier suitable for identifying the tagged section of the byte-code," and "the identified tagged section is compiled by the compiler when the byte-code is loaded so as to enable the tagged section of byte-code to be utilized without additional compiling of the tagged section of byte-code." The combination of these three limitations is not taught or suggested by Blandy. Blandy discloses a Just in Time station 224, a compiler 210, and an interpreter 212. Brandy teaches Just in Time compiling and interpreting, but not pre-compiling and not that "the identified tagged section is compiled by the compiler when the byte-code is loaded." The claims recited that the tagged section is identified by an identifier. Blandy, column 4, line 54, through column 5, line 23, discloses a compiler lock 227 which is used to serialize compilation of a method, an invoker field 228 of method block 226 that is set to point to the Just In Time initialization code, and branch monitors that may be implemented using breakpoints with interrupt handlers (i.e., through the operation of assembler language conditional jump instructions, as shown in column 5, lines 26-29). No identifier that identifies a tagged section of bytecode that is compiled when loaded is disclosed or suggested by Blandy. Thus, Blandy does not anticipate or make obvious Claims 13-17 and 35.

Moreover, even if Blandy were modifiable by Coutts, Coutts does not teach or suggest the combination of these limitations and does not remedy the deficiencies of Blandy.

Thus, Claims 13-17 and 35 are allowable over the prior art of record.

Claims 14-16 recite "an encoder for encoding the source code to byte-code" and "a tagger for tagging a section of the byte-code." Blandy does not teach or suggest a tagger for tagging a section of the byte-code. Figure 6 of Blandy does not disclose encoding; instead, Figure 6 illustrates a process for a fall through monitor. Blandy, column 5, lines 14-23, disclose breakpoints corresponding to assembler language jump instructions. The jump instructions of Blandy refer to byte-code and not to a tagger that tags a section of the byte-code. Thus, Blandy does not teach or disclose a tagger or an encoder.

Even if Blandy were modifiable by Coutts, Coutts does not teach or suggest the combination of these limitations.

Thus, Claims 14-16 are allowable over the prior art of record.

Claim 18 recites "a method for providing an execution environment in an information appliance network, comprising: a) encoding an application source code in a processor independent byte-code; b) tagging at least some portion of said processor independent byte-code; and c) compiling at least some portion of said tagged processor independent byte-code, wherein the independent byte-code includes at least one dynamic base object, the at least one dynamic base object comprising an interface dynamic base object and an implementation dynamic base object that communicate with each other over a message bus." In the present application, on page 9, lines 23-26, interface and implementation DBOs are described such that "when an application creates a DBO, two DBOs are actually created (FIGS. 6A and 6B). These two DBOs are an interface-DBO within the application, and an instance of the real DBO (a/k/a an implementation-DBO)." On page 10, lines 3-5, the present specification further describes interface and implementation DBOs "In a preferred embodiment of the invention, each time the application uses the interface-DBO, a message is sent to the implementation-DBO, which carries out the task and returns the result. When the application frees the DBO the reverse happens." Blandy, as recognized by the Patent Office, does not teach "at least

one dynamic base object, the at least one dynamic base object comprising an interface dynamic base object and an implementation dynamic base object that communicate with each other over a message bus." Blandy shows a block diagram of a data processing system in Figure 1. Blandy shows a Java virtual machine in Figure 2. Blandy (column 1, lines 50-61) discloses Java bytecode may be stored as a Java application or servlet, (column 2, lines 1-5) refers to Just in Timing compiling, and (column 3, lines 45-54) discloses data processing system 100 may be a standalone system. None of the figures or sections of Blandy cited by the Patent Office relate to a dynamic base object. Coutts (column 25, line 66, through column 26, line 10) discloses that Java byte codes may be executed directly in silicon. Coutts also discloses that using interpreters yields poor performance and Just In Time compilers require increased amounts of memory. Coutts shows a functional block diagram of a thin client application architecture in Figure 32. Coutts discloses (column 27, lines 26-50) a thin client used in point of sale applications.

The Final Office Action, on pages 4 and 5, furthermore asserts that Blandy and Coutts suggest "wherein the independent byte-code includes at least one dynamic base object, the at least one dynamic base object comprising an interface dynamic base object and an implementation dynamic base object that communicate with each other over a message bus." However, neither Blandy nor Coutts disclose or suggest this limitation. Applicant shows the relationship between the implementation DBO and interface DBO in Figures 4 and 6A of Applicant's application for patent (as reproduced below). There is no disclosure or suggestion in Blandy suggesting a need or desire for a "dynamic base object that includes an interface dynamic base object and an implementation dynamic base object that communicate with each other over a message bus." Obviousness cannot be established by combining the teachings of the prior art to produce the claimed invention, absent some teaching or suggestion supporting the combination. Under section 103, teachings of references can be combined only if there is some suggestion or incentive to do so. *ACS Hosp. Sys., Inc. v. Montefiore Hosp.,* 732 F.2d 1572, 221 USPQ 929 (Fed. Cir. 1984). Thus, the Examiner may not use the patent application as a basis for the motivation to combine or modify the prior art to arrive at the claimed invention. Thus, Claim 18 is allowable over the prior art of record.

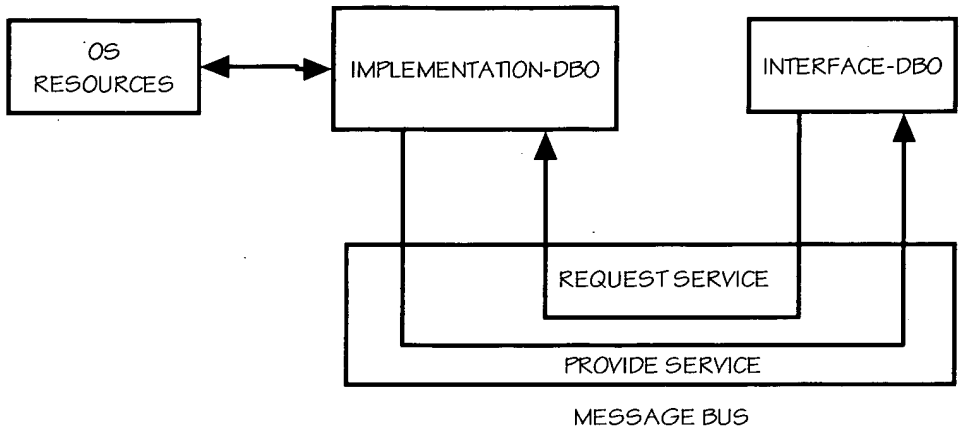
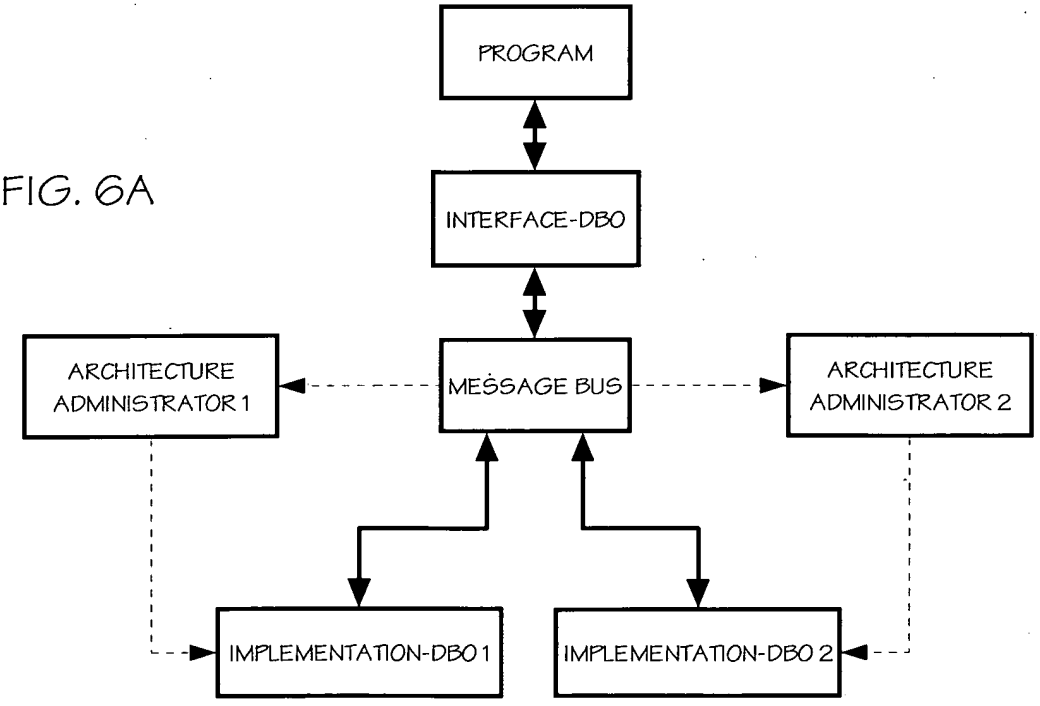

OS RESOURCES
IMPLEMENTATION-DBO
INTERFACE-DBO
REQUEST SERVICE
PROVIDE SERVICE
MESSAGE BUS

FIG. 4

FIG. 6A


PROGRAM
INTERFACE-DBO
ARCHITECTURE ADMINISTRATOR 1
MESSAGE BUS
ARCHITECTURE ADMINISTRATOR 2
IMPLEMENTATION-DBO 1
IMPLEMENTATION-DBO 2

Claim 20 recites "compiling includes identifying the portion of said tagged processor independent byte-code." Blandy, column 4, line 54, through column 5, line 23, discloses a compiler lock 227 which is used to serialize compilation of a method, an invoker field 228 of method block 226 that is set to point to the Just In Time initialization code, and branch monitors that may be implemented using breakpoints with interrupt handlers (i.e., through the operation of assembler language conditional jump instructions, as shown in column 5, lines 26-29). No identifier that identifies a tagged section of bytecode that is compiled when loaded is disclosed or suggested by Blandy. Coutts does not teach or suggest these limitations. As the combination of Blandy and Coutts does not make obvious Claim 20, Claim 20 is allowable over the prior art of record for this reason as well as being dependent upon allowable Claim 18.

Claim 1-12, 21-34, and 36-44 recite "a method for dynamic compiling" that includes loading byte-code on a digital information appliance, said byte-code suitable for including a tagged section," "identifying the tagged section of the byte-code," "wherein the tagged section is compiled when the byte-code is loaded so as to enable the digital information appliance to utilize the tagged section of byte-code without additional compiling of the tagged section of byte-code by the digital information appliance," and "wherein the byte-code includes at least one dynamic base object, the at least one dynamic base object comprising an interface dynamic base object and an implementation dynamic base object that communicate with each other over a message bus." Blandy does not teach compiling the tagged section when the byte-code is loaded. The Patent Office cited column 3, lines 24-28, and column 4, lines 7-16, of Blandy as teaching this limitation. The first cited portion of Blandy on column 3, lines 24-28, asserts "Instructions for the operating system, the object-oriented operating system, and applications or programs are located on storage devices, such as hard disk drive 126, and may be loaded into main memory 104 for execution by processor 102" and does not teach or suggest that the tagged section is compiled when loaded. Any tagged section of the byte-code could be compiled before loading or after loading. The second cited portion of Blandy on column 4, lines 7-16, discloses a Just in Time compiler 210, an interpreter 212 and bytecodes described as a sequence of instructions, does not teach or suggest that the

tagged section is compiled when loaded. Just In Time compiling refers to compilation immediately before execution. An interpreter translates and runs code at the same time. Coutts does not remedy the deficiencies of Blandy. Thus, Claims 1-12, 21-34, and 36-44 are not made obvious by Blandy in view of Coutts. Furthermore, as Claims 1-12, 21-34, and 36-44 have limitations found in both Claims 13-17 and 18-20, the combination of the limitations of these claims is believed to be novel and non-obvious. Therefore, Claims 1-12, 21-34, and 36-44 are allowable over the prior art of record.

Claims 2 and 8 recite "encoding application source code to byte-code, the byte-code including code in a processor-independent form which is suitable for further analysis and tagging a section of the byte-code." Figure 13 (see below) illustrates the process recited in claims 2 and 8. Blandy discloses (Figure 4, column 5, lines 14-23) jump points and breakpoints already found in the byte code and does not teach encoding the application code to byte-code and tagging a section of the byte-code. Blandy does not teach or suggest the limitations of Claims 2 or 8. Coutts is directed to a transaction processing system that uses a Java Virtual Machine built in silicon rather than use a Just In Time compiler or interpreter. Thus, Claims 2 and 8 are allowable over the prior art of record for this additional reason.

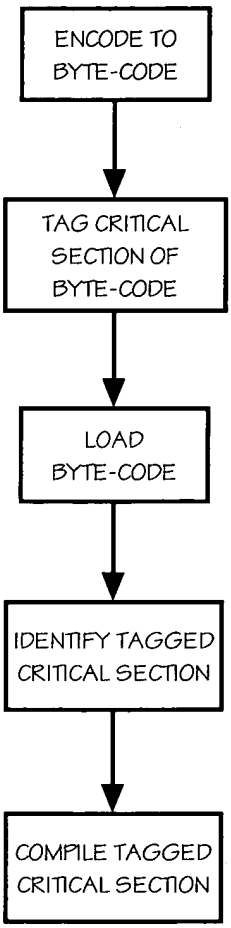


FIG. 13

Claims 24, 26, 36, and 40 recite "wherein the dynamic base object is programmed through a scripting language with run-time object invocation." Coutts, as discussed above does not disclose a dynamic base object. Although Coutts (column 11, lines 44-49) does disclose reports in HTML form, Coutts does not disclose a scripting language for programming a dynamic base object. As discussed regarding the rejection of Claim 18, neither Blandy nor Coutts teach or suggest "at least one dynamic base object comprising an interface dynamic base object and an implementation dynamic base object that communicate with each other over a message bus," and so do not teach or suggest a

dynamic base object programmed through a scripting language with run-time object invocation." Thus, Claims 24, 26, 36, and 40 are allowable over the prior art of record for this additional reason.

Claims 25, 30, 34, and 41 recite "the interface dynamic base object and the implementation dynamic base object communicate bi-directionally." Coutts (Figures 14-16, 30, 32) show transaction networks and terminals, but does not show at least one dynamic base object that has an interface dynamic base object and an implementation dynamic base object that communicate bi-directionally. Thus, Claims 25, 30, 34, and 41 are allowable over the prior art of record for this additional reason.

Claim 44 recites "wherein the at least one dynamic base object is fully thread safe." That is, the at least one dynamic base object may be utilized by concurrent multiple thread applications. Coutts (column 4, lines 54-59) discloses the compiler lock prevents a method from being compiled by multithreads. Thus, Claim 44 is allowable over the prior art of record.

The Patent Office rejected Claims 6, 12, 17, and 19 under 35 U.S.C. 103(a) as being unpatentable over Blandy, U.S. Patent No. 6,295,642, and Coutts, et al., U.S. Patent No. 6,311,165, as applied to Claims 1, 7, 13, 18, and further in view of Hamby et al., U.S. Patent No. 5,848,274.

Claim 17 recites "the loader includes a validation utility for validating the byte-code conforms with byte-code suitable for utilization by the system." Claims 6, 12, and 19 similarly recite validating the byte-code. Blandy does not disclose a validation utility for validating the byte-code conforms with byte-code suitable for utilization by the system. Blandy, on column 4, lines 54-59, ensures that the same method is not compiled simultaneously by multiple threads and does not teach or suggest "a validation utility for validating the byte-code conforms with byte-code suitable for utilization by the system." No basis has been provided for rejecting Claim 17 by the combination of Blandy and Coutts because Coutts has been cited by the Patent Office for a teaching about dynamic

base objects – a limitation not present in Claim 17. Even if Blandy were modifiable by Coutts, Coutts does not teach or suggest a validation utility. Coutts, on column 44, lines 19-39, is directed to messages and not byte-code. Hamby, on column 27, line 42, through column 28, line 6, discloses an incremental byte compiler, but does not disclose "a validation utility for validating the byte-code conforms with byte-code suitable for utilization by the system." Interrogation of the operating system to find a file (col. 28, lines 7-8, Hamby) is not validation that the byte-code conforms with byte-code suitable for utilization by the system." The last six lines of page 11 and the first two lines of page 12 of the Final Office Action asserting byte-code validation is not a fair reading of Blandy, Coutts, and/or Hamby, and does not conform to the cited passages from those three references. Thus, Claims 6, 12, 17, and 19 are allowable over the prior art on their own merit as well as their dependency from base claims 1, 7, 13, and 18.

CONCLUSION

In light of the foregoing, amendments and supporting arguments, reconsideration of all pending claims is requested, and a Notice of Allowance is earnestly solicited.

Respectfully submitted,

GATEWAY, INC.,

By: Walter J. Malinowski

Walter Malinowski
Reg. No. 43,423

Suiter West pc llo
14301 FNB Parkway, Suite 220
Omaha, Nebraska 68154
Telephone 402.496.0300
Facsimile 402.496.0333